

B. TECH. PROJECT REPORT

On

Developing a cost-effective Reliability Management tool

BY

Suraj Yadav



**DISCIPLINE OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE**

Developing a cost-effective reliability management tool

A PROJECT REPORT

*Submitted in partial fulfillment of the
requirements for the award of the degrees*

of

BACHELOR OF TECHNOLOGY

In

MECHANICAL ENGINEERING

Submitted by:

SURAJ YADAV

Guided by:

Professor. Bhupesh Kumar Lad

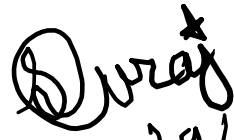
INDIAN INSTITUTE OF TECHNOLOGY INDORE

November 2023

CANDIDATE'S DECLARATION

We hereby declare that the project entitled “**Developing a cost-effective reliability management tool**” submitted in partial fulfillment for the award of the degree of Bachelor of Technology in **Mechanical Engineering** completed under the supervision of **Prof. Bhupesh Kumar Lad**, IIT Indore is an authentic work.

Further, I/we declare that I/we have not submitted this work for the award of any other degree elsewhere.



29/11/23

Suraj Yadav

Signature and name of the student(s) with date

CERTIFICATE by BTP Guide(s)

It is certified that the above statement made by the students is correct to the best of my/our knowledge.


29/11/23 Professor

Signature of BTP Guide(s) with dates and their designation

Preface

This report on “Cost effective reliability management tool” is prepared under the guidance of Dr Bhupesh Kumar Lad, Professor, IIT Indore.

Through this report I have tried to give a detailed explanation of how we designed and developed a cost-effective solution of a reliability management tool. I have tried to explain the complex functions and methods that are used in our webtool in a concise and lucid way to the best of my abilities. I have also added associated graphs and images for a clear understanding of the topic and the problem statement.

The report further explains the usage of the webtool and also the advantages of using a web-based tool deployed on the cloud over traditional in PC installations of the software in terms of cost and quick accessibility.

Suraj Yadav

B.Tech. IV Year

Discipline of Mechanical Engineering

IIT Indore

Acknowledgements

I wish to thank Dr Bhupesh Kumar Lad, Professor for his kind support and valuable guidance, without whom it would not have been possible to complete this project. I would also like to thank PHD student Ram Mohril and the IITI Drishti team who helped me understand the framework and technology associated with reliability.

It is their help and support, due to which I was able to complete the design and technical report.

Without their support this report would not have been possible.

Suraj Yadav

B.Tech. IV Year

Discipline of Mechanical Engineering

IIT Indore

Abstract

The problem statement of the project is to develop a cost-effective reliability management webtool that is easily accessible by users and highly compatible with most devices. Present software's available in the market face two major challenges, firstly software compatibility issues arise during installation of the software and secondly, they are expensive to use.

A solution that is implemented in this project is to use the software as a service model of cloud computing which ensures easy accessibility. Since the solution is deployed on the web, it eradicates device compatibility issues, it is also based on a pay to use model which solves the economic challenges associated. The features of the webtool involve Life data analysis which deals with accurately computing reliability metrics, knowing which distribution pattern fits the best according to failure data, and performing subsequent calculations based on selected distribution including calculation of mean life, conditional probabilities of failure, plots of reliability vs time etc. given the failure data of a component. Another commonly used feature in the market is Block Simulation, which involves calculating system reliability metrics given individual component data and the configuration in which they are connected. Configurations involve simple series, parallel combination of components to form a system. The project involves usage of web technologies such as HTML, CSS, JavaScript, Flask backend framework and cloud computing. Reliability estimation tools are always in high demand as industries always want to improve the reliability of their products.

Table of Contents

CANDIDATE’S DECLARATION	3
Preface.....	4
Acknowledgements.....	5
Abstract.....	6
Chapter 1 - Introduction	11
1.1 Reliability	11
1.2 Challenges with present reliability estimation tools	11
1.3 Solution approach.....	12
Chapter 2 - Fundamentals of Reliability.....	13
2.1 Reliability Engineering	13
2.2 Failure Distributions	15
2.3 Reliability Metrics.....	18
Chapter 3 - Analysis Techniques	20
3.1 Life data analysis (LDA).....	20
3.2 Analytical approach for multi-component systems.....	24
3.2.1 Series RBD.....	24

3.2.2 Parallel RBD	25
3.2.3 Mixed RBD	25
Chapter 4 - Webtool design.....	26
4.1 Features implemented for single component systems.....	26
4.1.1 Handling different types of Failure data:	26
4.1.2 Selection of different distributions:.....	26
4.1.3 Quick calculation tools for reliability estimation.....	26
4.2 Features implemented for multi component systems.....	26
4.3 Technology used	27
4.3.1 Frameworks and languages	27
4.3.2 Existing Libraries used.....	28
4.4 Flow of the web tool for single component systems.....	29
Chapter 5 - Backend Development.....	30
5.1 Server-side logic.....	30
5.2 Programming logic for system reliability	32
5.2.1 Solvable Configurations.....	32
5.2.2 Limitations and challenges of the system:	34
Chapter 6 - Demonstration of web tool using sample data	35
6.1 Life data features.....	35
6.1.1 Entering data	35
6.1.2 Distribution selection and ranking	36

6.1.3 Parameters page	36
6.1.4 Quick calculation pad.....	37
6.2 Block sim backend features	39
6.2.1 Entering node data.....	39
6.2.2 Define connections number and pair wise names of blocks	39
6.2.3 Output system at particular time.	40
Chapter 7 - Conclusion and future scope.....	40
7.1 Results	40
7.2 Future Scope	41
References	42

List of Figures

Figure 1: Weibull PDF plot.....	15
Figure 2 Exponential survival function.....	16
Figure 3 Series RBD	24
Figure 4 Parallel Config.....	25
Figure 5 Data Flow	30
Figure 6 Fetch API calls.....	31
Figure 7 Exact data input frontend.....	35
Figure 8- distribution ranking page.....	36
Figure 9 Parameters page	36
Figure 10 Qcp-plots	37
Figure 11 Qcp-Bx life	37
Figure 12 Qcp-reliability.....	38
Figure 13 Qcp mean-life	38
Figure 14 Block sim Terminal input	39
Figure 15 Block sim connections	39
Figure 16 RBD config-1	41
Figure 17RBD config -2	41

Chapter 1 - Introduction

1.1 Reliability

Reliability refers to the ability of a system, component, or product to perform its intended function without failure over a specified period of time and under specified operating conditions. It is a crucial aspect of engineering design and manufacturing, particularly in fields such as aerospace, automotive and other industries where the performance and safety of products are paramount.

Given the significance of reliability in engineering sectors, there is a heightened demand for reliability estimation tools in the market.

1.2 Challenges with present reliability estimation tools

The current reliability estimation tools in the market encounter two primary challenges.

- Firstly, their pricing model is notably high, as they charge on an annual basis without considering the frequency of software usage.
- Secondly, these tools have specific system requirements, and failure to meet these prerequisites hinders usage, thereby imposing additional costs on the buyer to meet those specifications.
- Thirdly, the onsite installation services are also a challenge related to time and cost.

1.3 Solution approach

To overcome these challenges, a solution can be developed that leverages cloud technology. By creating a cloud-based tool, users can access it easily without the need for specific system requirements. This tool follows a pay-as-you-go model, ensuring that users only pay for the software when they are actively using it, thus eliminating costs associated with idle software time.

The cloud-based approach also offers scalability, allowing the tool to handle a large number of users effectively. Computing resources can be adjusted as needed, ensuring optimal performance and accommodating varying user demands.

The development process involves designing and implementing the necessary features for handling different types of component data on the tool's backend. This includes considering both single component systems and multicomponent systems to provide comprehensive analysis capabilities. To enhance user experience and ease of interaction, a frontend interface is also designed and developed. The frontend and backend are then seamlessly integrated to create a unified and functional tool.

To meet the demands of users, the tool should include key features such as life data analysis and effective system reliability analysis. These features are highly sought after and contribute to the tool's usefulness and effectiveness in assessing and evaluating the reliability of systems.

In summary, addressing the challenges outlined involves the development of a cloud-based tool that follows a pay-as-you-go model. This approach eliminates idle software costs, ensures accessibility regardless of system requirements, and offers scalability. The tool encompasses a comprehensive range of features for analyzing both single component and multicomponent systems, including popular capabilities like life data analysis and effective system reliability analysis. By incorporating these elements, the tool can provide valuable insights and support decision-making processes related to system reliability.

Chapter 2 - Fundamentals of Reliability

2.1 Reliability Engineering

Reliability engineering is a specialized discipline within the field of engineering that concentrates on the design, implementation, and upkeep of systems, products, and processes to guarantee a high level of reliability. In this context, reliability pertains to the capacity of a system or component to fulfill its intended function without failure within specified timeframes and under predetermined conditions.

Reliability engineering analysis plays a crucial role in improving product quality by identifying potential issues, weaknesses, and failure modes in the design and manufacturing processes.

For Example, Life data analysis, a well know method of analysis in reliability engineering helps industries in quality improvement in the following ways:

1. **Reliability Assessment:** Life Data Analysis allows engineers to assess the reliability of a product by analysing the time-to-failure data. This helps in understanding the distribution of failure times and estimating key reliability metrics, such as mean time to failure (MTTF) or mean time between failures (MTBF). This information is crucial for predicting and improving product reliability.
2. **Identifying Failure Patterns:** LDA helps identify patterns in failure occurrences. Engineers can determine whether failures follow a specific distribution (e.g., exponential, Weibull) and gain insights into the predominant failure modes. This understanding guides improvements in design, materials, and manufacturing processes to address the identified failure patterns.
3. **Predictive Modelling:** Life Data Analysis allows for the creation of predictive models based on the observed failure data. By fitting statistical distributions to the failure data, engineers can develop models that predict the probability of failure over time. These models aid in making informed decisions about maintenance intervals, warranty periods, and product lifecycle planning.

4. **Optimizing Preventive Maintenance:** With insights gained from Life Data Analysis, companies can optimize preventive maintenance strategies. By understanding the expected failure patterns and probabilities, maintenance activities can be scheduled proactively, minimizing downtime and reducing maintenance costs.
5. **Quality Improvement During Prototyping:** Life Data Analysis can be applied during the prototyping phase to evaluate the reliability of early versions of a product. This allows engineers to identify and address potential reliability issues before mass production, reducing the likelihood of defects and improving overall product quality.
6. **Design Optimization:** By analysing life data, engineers can optimize product design for reliability. This includes selecting materials, components, and manufacturing processes that contribute to longer product life and improved durability. Design decisions can be informed by insights gained from Life Data Analysis.
7. **Customer Satisfaction:** Products that consistently meet or exceed reliability expectations contribute to customer satisfaction. Life Data Analysis helps ensure that products perform reliably over their expected lifespan, leading to positive customer experiences and building trust in the brand.

2.2 Failure Distributions

In the field of reliability engineering, various failure distributions are available and these distributions help us in determining the probability of failure of the any component or machine. The following are the distributions employed in this particular project.

▪ **Weibull Distribution:**

Widely used distribution in modelling failure probability in the market. This distribution can be further classified into 2 sub types based on the number of parameters.

- **Weibull 2P:**

- ◆ **Formula:** $R(t) = e^{-\left(\frac{t}{\alpha}\right)^\beta}$

- ◆ **Parameters:** The two parameters are the scale parameter (α or η) and the shape parameter (β).

- ◆ **Use Cases:** The two-parameter Weibull distribution is often used when the primary focus is on modelling reliability and survival data. It's suitable for situations where the failure rate is not constant over time. Commonly employed in reliability engineering to analyse time-to-failure data and estimate the reliability of systems.

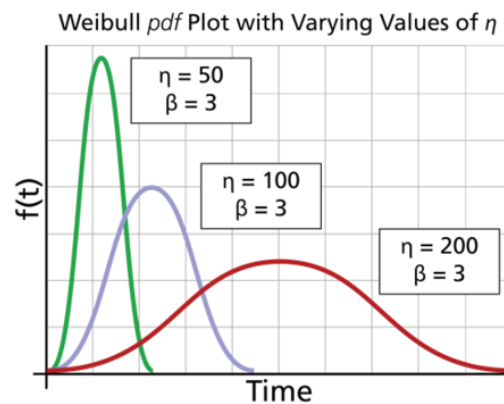


Figure 1: Weibull PDF plot

- **Weibull 3P:**

- ♦ **Parameters:** The three parameters are the scale parameter (α), the shape parameter (β), and the location parameter (γ).
- ♦ **Use Cases:** Flexibility in Modeling, the three-parameter version adds a location parameter, allowing for greater flexibility in modeling data with nonzero lower bounds. It is useful when there is a need to shift or translate the distribution along the x-axis.

- **Exponential Distribution:**

- ♦ **Formula:**

$$R(t) = e^{-\lambda t}$$

- ♦ **Parameters:** Lambda (λ) the scale parameter

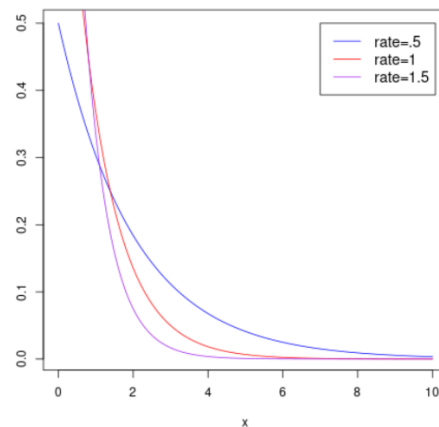


Figure 2 Exponential survival function

- **Normal Distribution**

- ♦ **Formula:**

$$R(t) = 1 - \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^t \exp \left[-\frac{1}{2} \left(\frac{\theta - \mu}{\sigma} \right)^2 \right] d\theta$$

- ♦ **Parameters:** Mu (μ) the location parameter and sigma (σ) the shape parameter

▪ Gamma Distribution

♦ Formula:

$$R(t) = \frac{1}{\Gamma(\beta)} \Gamma\left(\beta, \frac{t}{\alpha}\right)$$

- ♦ **Parameters:** The two parameters are the scale parameter (α or η) and the shape parameter (β).

▪ Gumbel Distribution

♦ Formula:

$$R(t) = e^{-e^z}$$

$$\text{where } z = \frac{t-\mu}{\sigma}$$

- ♦ **Parameters:** Mu (μ) the location parameter and sigma (σ) the shape parameter

▪ Loglogistic Distribution

♦ Formula:

$$R(t) = \frac{1}{1 + \left(\frac{t}{\alpha}\right)^\beta}$$

- ♦ **Parameters:** The two parameters are the scale parameter (α or η) and the shape parameter (β).

▪ Lognormal

♦ Formula:

$$R(t) = 1 - \Phi\left(\frac{\ln(t)-\mu}{\sigma}\right)$$

where Φ is the standard normal CDF with $\mu = 0$ and $\sigma = 1$

Parameters: Mu (μ) the location parameter and sigma (σ) the shape parameter

2.3 Reliability Metrics

Reliability metrics are measures used to assess and quantify the performance and dependability of systems, products, or processes. These metrics provide insights into the likelihood of failures, the availability of a system, and overall performance. Several metrics can be derived from the failure distribution itself such as the survival function, cumulative density function, etc. Apart from these there are other metrics such as the following which have been incorporated in this project.

➤ **Mean Life:**

- In the present context of failure data analysis mean life is defined as the time taken for the component reliability to become 50 %. MTTF on the other represents the mean life to failure which is the average time taken for the component failure and is calculated analytically via integrating the probability density function.

➤ **BX life:**

- This represents the point in time at which a certain percentage, denoted by "X," of the items in a population have failed. For example, if you say "B10 life," it means the point at which 10% of the items have failed. BX% life is often used in reliability testing and estimation to understand the performance and durability of products or systems.
- Reliability curves, such as the Weibull reliability function, are used to model the failure distribution of items over time, and BX% life can be determined from these curves.

➤ **Probability of failure:**

- The probability that a component fails at a given time is defined as probability of failure.
- Can mathematically be computed by $1 - \text{reliability of the component at time 't'}$.

➤ **Conditional reliability:**

- Conditional reliability refers to the probability that a component survives a given mission time of 't' given that it has survived a mission time of 'T'. Mathematically can be represented as follows:

$$R(t|T) = \frac{R(t + T)}{R(T)}$$

- **Conditional probability of failure:** The probability that a component fails a given mission time 't' given it has survived a given that is survived a mission time of 'T'. mathematically it is $1 - \text{conditional reliability } (t, T)$.

- **Reliable life:** It gives the time for which the component reliability is above a given threshold of reliability. For example, reliable life at $r=75\%$ would mean the maximum time till the reliability of the component is above 75%. (Inverse of the survival function at $r = 0.75$).

Chapter 3 - Analysis Techniques

3.1 Life data analysis (LDA)

Life Data Analysis is a statistical method used in reliability engineering to analyze and interpret data related to the lifetimes or failure times of components or systems. It involves the study of time-to-failure data to make predictions about the reliability and performance of a product or system.

We do this by trying to fit the failure data to cumulative distribution function of any of the available failure distributions such a Weibull, exponential etc and ranking their order of fit based on error. Deciding which distribution is the best fit can be done via error analysis techniques out of which the most commonly used are Least square estimation (LSE) and Maximum likelihood estimation (MLE). We determine the best distribution and their corresponding parameters through these techniques and then use the survival functions of these distributions to estimate the reliability of the component or system.

Least Square Estimation:

Least Squares Estimation (LSE) is a statistical method used to estimate the parameters of a model by minimizing the sum of the squared differences between observed and predicted values. This approach is particularly common in linear regression analysis, where the goal is to find the best-fitting line through a set of data points. In case of failure distributions, the equations are first converted into linear terms by applying the logarithmic functions on both sides and then applying least square estimation to find the parameters of the best fit.

Let use apply LSE for the CDF of Weibull distribution given below

k = shape parameter

c = scale parameter

$$F(v) = 1 - \exp\left[-\left(\frac{v}{c}\right)^k\right]$$

But the above expression can be expressed linearly as

$$\ln \ln \left\{ \frac{1}{1-F(v)} \right\} = k \ln v - k \ln c. \quad (\text{Exp-2})$$

Exp 2 can be compared with $Y = bX + a$;

Where,

$$Y = \ln \ln \left\{ \frac{1}{1-F(v)} \right\}, \quad X = \ln v, \quad a = -k \ln c, \quad b = k$$

Solving a linear regression problem then becomes a rather simplified task, we must minimise the sum of square of difference of the actual value and the function value, that is

$$S = \sum_{i=1}^n w_i^2 [y_i - g(x_i)]^2$$

Where, y_i = actual value $g(x_i)$ = function output value

Solving for minimum error sum gives a and b values as follows

$$b = \frac{N \sum_{i=1}^n X_i Y_i - \sum_{i=1}^n X_i \sum_{i=1}^n Y_i}{N \sum_{i=1}^n X_i^2 - \left(\sum_{i=1}^n X_i \right)^2}$$

$$a = \frac{\sum_{i=1}^n X_i^2 \sum_{i=1}^n Y_i - \sum_{i=1}^n X_i \sum_{i=1}^n X_i Y_i}{N \sum_{i=1}^n X_i^2 - \left(\sum_{i=1}^n X_i \right)^2}$$

Maximum likelihood estimation:

MLE is a statistical method used to estimate the parameters of a statistical model. The idea behind MLE is to find the values of the model parameters that maximize the likelihood function. The likelihood function represents the probability of observing the given data under the assumed statistical model.

It is common to work with the log-likelihood function (ℓ), which is the natural logarithm of the likelihood function. This is because the logarithm simplifies computations, and the maximum of the log-likelihood occurs at the same point as the maximum of the likelihood function. The objective of MLE is to find the values of the parameters that maximize the log-likelihood function.

Example of MLE in case of normal distribution

Suppose you have a set of independent and identically distributed observations X_1, X_2, \dots, X_n from a normal distribution with unknown mean μ and known standard deviation σ . The probability density function (PDF) of the normal distribution is given by:

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Likelihood Function:

The likelihood function for the observed data X_1, X_2, \dots, X_n is given by the product of the individual probabilities:

$$L(\mu, \sigma | X_1, X_2, \dots, X_n) = \prod f(x_i; \mu, \sigma)$$

Log-Likelihood Function:

To simplify computations, it's common to work with the log-likelihood function, which is the natural logarithm of the likelihood function n:

$$\ell(\mu, \sigma | X_1, X_2, \dots, X_n) = \sum_{i=1}^n \left[-\frac{1}{2} \ln(2\pi\sigma^2) - \frac{(x_i - \mu)^2}{2\sigma^2} \right]$$

$$\mu, \sigma = \arg\max \ell(\mu, \sigma | X_1, X_2, \dots, X_n)$$

Derivation:

To find the maximum likelihood estimates, take the partial derivatives of the log-likelihood function with respect to μ and σ , set them equal to zero, and solve for the parameters.

$$\partial \ell / \partial \sigma = 0 \quad \partial \ell / \partial \mu = 0$$

Result:

The solutions to the above equations give the maximum likelihood estimates as follows

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2}$$

3.2 Analytical approach for multi-component systems

For multi-component systems, a necessary reliability block diagram (RBD) is constructed, and subsequent analyses are conducted, including determining the system's reliability over time and computing reliability metrics like b10 life and mean life. When analyzing a system, it is crucial to have prior knowledge of the reliability distribution and parameters that best suit each individual component within the system.

Calculating the reliability of the entire system requires employing formulas for equivalent block diagrams, especially in the context of series and parallel connections. It is important to note that each block may also represent a subsystem, potentially comprising multiple components, each with distinct distributions and parameters.

3.2.1 Series RBD

Components connected in series, in such a block diagram failure of one component causes the system to fail hence the reliability of the system is the product of the individual reliabilities of the components.

$$R_{eq} = \prod_{i=1}^n R_i$$

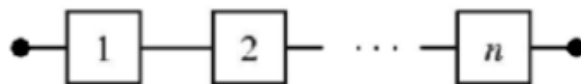


Figure 3 Series RBD

3.2.2 Parallel RBD

In parallel circuits, the failure of a specific component does not result in system failure. The entire system only fails if all components within the parallel circuit fail simultaneously. Therefore, the reliability can be determined as the probability that none of the components fail, as expressed by the formula:

$$R_{eq} = 1 - \prod_{i=1}^n (1 - R_i)$$

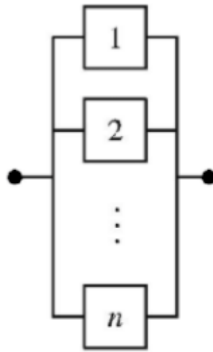
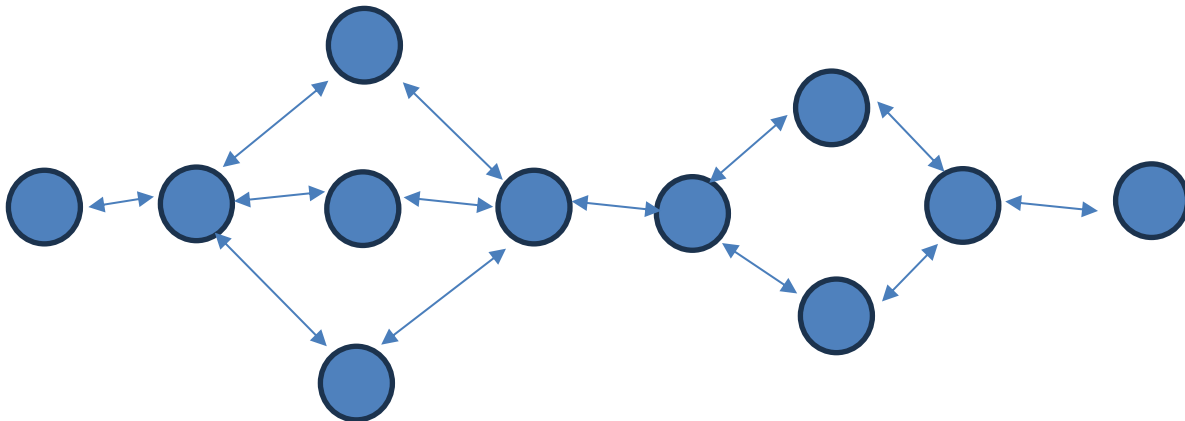


Figure 4 Parallel Config

3.2.3 Mixed RBD

A diagram consisting of both series and parallel circuits must be solved by reducing the diagram by identifying series and parallel components within the diagram.



Chapter 4 - Webtool design

4.1 Features implemented for single component systems

4.1.1 Handling different types of Failure data:

- Exact data: Exact failure time of the component is known
- Interval Data: A interval between which failure occurs is known.
- Right censored data: Last known survival time is known.

4.1.2 Selection of different distributions:

- The webtool allows users to select through a variety of failure distributions and outputs the ranking of fit.
- A table representing the parameters of the selected distribution according to ranking of best fit is also shown.

4.1.3 Quick calculation tools for reliability estimation

- A toolbar to select the distribution and output the resulting graph of reliability vs time, unreliability vs time and the pdf plot based on selection.
- Different options for BX life data, reliable life, conditional probability of failure, conditional reliability, Mean life etc.

4.2 Features implemented for multi component systems

- A python backend tool that can take input the reliability block diagram and process the RBD to calculate the system reliability as a function of time.

4.3 Technology used

4.3.1 Frameworks and languages

For the web interface or frontend, the project uses html 5, CSS, Java script and bootstrap for creating a user-friendly interface. Python as the backend server-side language and Flask framework to integrate the Front end and backend.

Flask Framework

Flask is a lightweight and flexible web framework for Python. It is designed to be simple and easy to use, making it a popular choice for building web applications and APIs and concepts related to Flask:

Routing:

- Flask uses a decorator-based syntax for defining routes. Routes map URLs to functions, allowing you to define what should happen when a user accesses a specific endpoint.

Templates:

- Flask uses the Jinja2 templating engine, allowing users to build dynamic HTML pages. Templates help separate the presentation logic from the application logic.
- Templating has several uses such as dynamically passing values and parameters to predefined templates for rendering dynamic web pages, this also helps in displaying plots that are rendered from user defined data.

Request and Response Handling:

- Flask provides convenient methods for handling incoming requests and generating responses, for example GET and POST requests can be handled as follows:

```
@app.route('/test', methods=['GET', 'POST'])
def sample_post_get():
    if request.method == 'POST':
        return 'Hello, ' + request.form['name']
```

4.3.2 Existing Libraries used

➤ Matplotlib

- Matplotlib is a 2D plotting library for Python. It allows you to create static, animated, and interactive visualizations in Python.
- Key Features:
 - Line plots, scatter plots, bar plots, histograms, pie charts, etc.
 - Customizable plots with various styling options.

➤ Flask sessions

- Flask Sessions refer to a mechanism in the Flask web framework for persisting user-specific data across requests. It allows the storage and retrieval of data on a per-session basis.
- Key Features:
 - Enables the storage of user-specific information, such as login state or user preferences.
 - Utilizes cookies or server-side storage to maintain session data.
 - Helps in building stateful web applications.

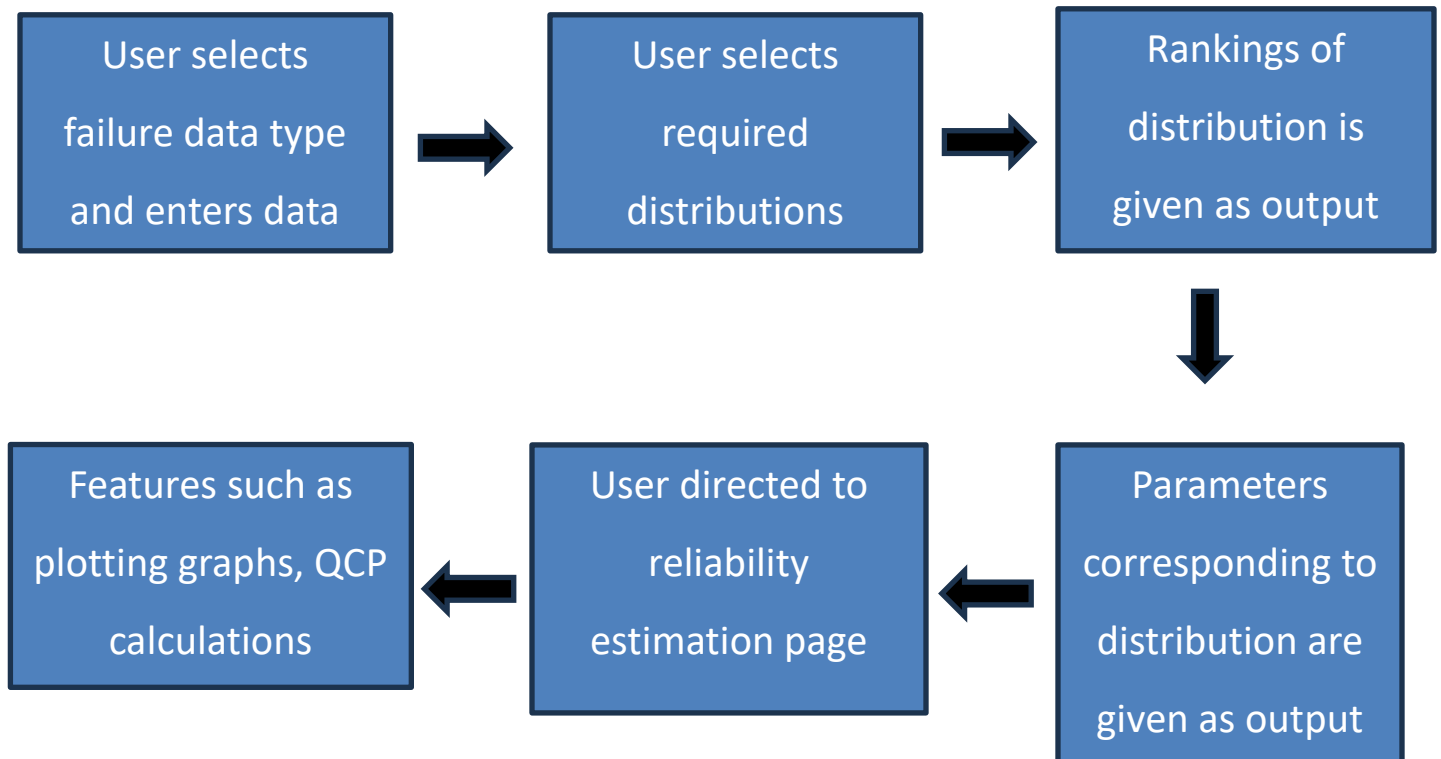
➤ IO

- The io module comes with several classes and functions to deal with different types of I/O, such as file-like objects, string-like objects, and more. Here are some key aspects of the io module:
 - The io module includes Bytes IO and String IO classes that allow you to treat binary data and strings as file-like objects. This can be incredibly useful especially in case of file or image transfer between server and client dynamically.

➤ Reliability Library

- It consists of several function useful in the context of reliability engineering such as formulae of failure distributions.
- Give BIC (Bayer information criterion) values for fit of distribution according to data.
- Plots of several complex reliability equations.

4.4 Flow of the web tool for single component systems



Chapter 5 - Backend Development

5.1 Server-side logic

The backend is built completely on the request-response model, which is for a given request on the webpage the python webserver return a response. The response could be anything such as another html page, Json response etc. This is managed by the flask framework.

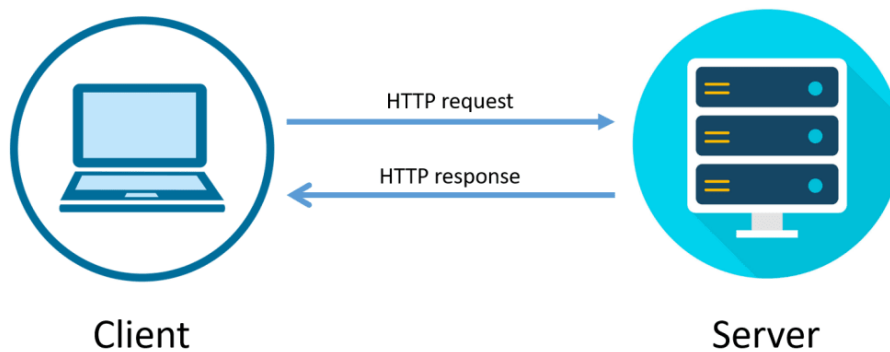


Figure 5 Data Flow

First the user goes to the main page where he logs in and moves to the simulation section where he selects his data type as an html corresponding to the selected data type is returned as response.

After the user enters the data into the frontend that is the failure times and clicks on submit, the “inputSubmit” function is triggered at the backend which stores the given data as a user state in flask sessions and returns the distribution selection page. It is important to note the failure data is not stored in a database but using **flask sessions** in the cache memory.

Following this the distributions are displayed on the screen using the html form tag which allow the user to select the distribution he wants to work with. On clicking the submit button the “distSelectSubmit” function is called that calls the “find_best” function in the helper.py file. This function utilizes the reliability library to get the BIC values for each distribution selected according to the failure data, the function further sorts the values based on increasing values of the BIC as a smaller value of BIC indicates a better fit. The sorted array is zipped with the BIC value and the distribution and returned. The returned array is then further processed

so that only the distributions names are available in the array and the array is passed to the response html page as parameters using **jinja templating**.

Next the user clicks on the parameters page button and the response here is handled by the “parameters ()” function. This function calls the “get_para_data” function forms the helper.py file. The “get_para_data” function gets the corresponding parameter values which have been calculated via the MLE method in the reliability library are returned in the form of an array. This array is hen passed as a parameter to the frontend again via jinja Templating.

The user further clicks on the reliability estimation page button where he is redirected to the Quick calculation pad for further calculations. Here the user can select the distribution he wants to continue with and choose options corresponding to calculation of reliability metrics.

There are 2 main options, The Probability options which consist of reliability, probability of failure, conditional reliability, condition probability of failure and Life Analysis options which consist of BX% life, reliable life and mean life. Each option has its own function that is called at backend, but these options call the functions via JavaScript using the Fetch API to send to and fro data as the page does not reload and the results are rendered using JavaScript.

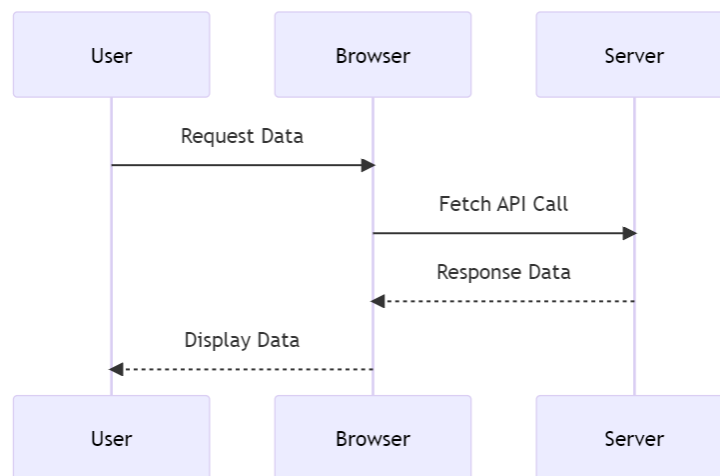


Figure 6 Fetch API calls

The plots are displayed by converting the generated graph to byte 64 format and sent as response using the send_file library and accessed on the client side via the fetch Api and rendered by JavaScript.

5.2 Programming logic for system reliability

Computing system reliability for a human on paper can seem easy but to explain the logic to a computer can be a challenging task, the logic used in the project to calculate system reliability is as follows:

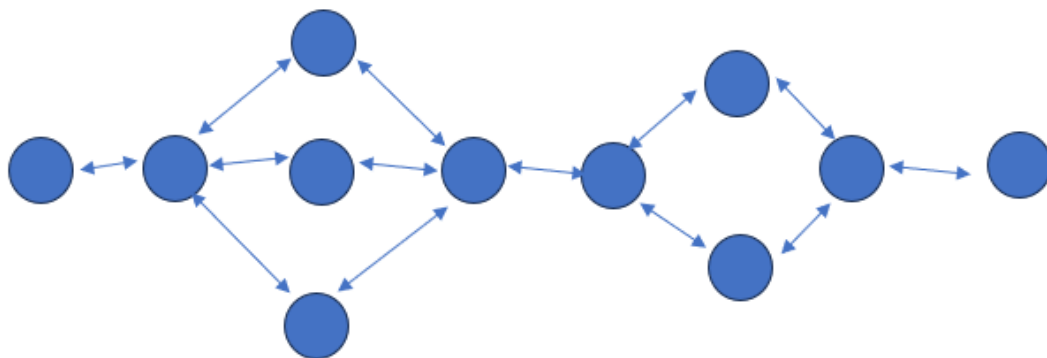
With a simple observation it can be seen that all circuits that are solvable will consist of a section which is a combination of simple series parallel which means purely series circuits and purely parallel circuits will be connected in series. Hence if we can solve this subproblem we can use recursion to solve the entire circuit.

In the code I have done just that, for complex parts that is connections that are not purely series and parallel the user himself marks those regions as complex and the algorithm uses recursion to solve the system. As we can see the system has certain user dependency but this can be removed by further optimizing the algorithm.

5.2.1 Solvable Configurations

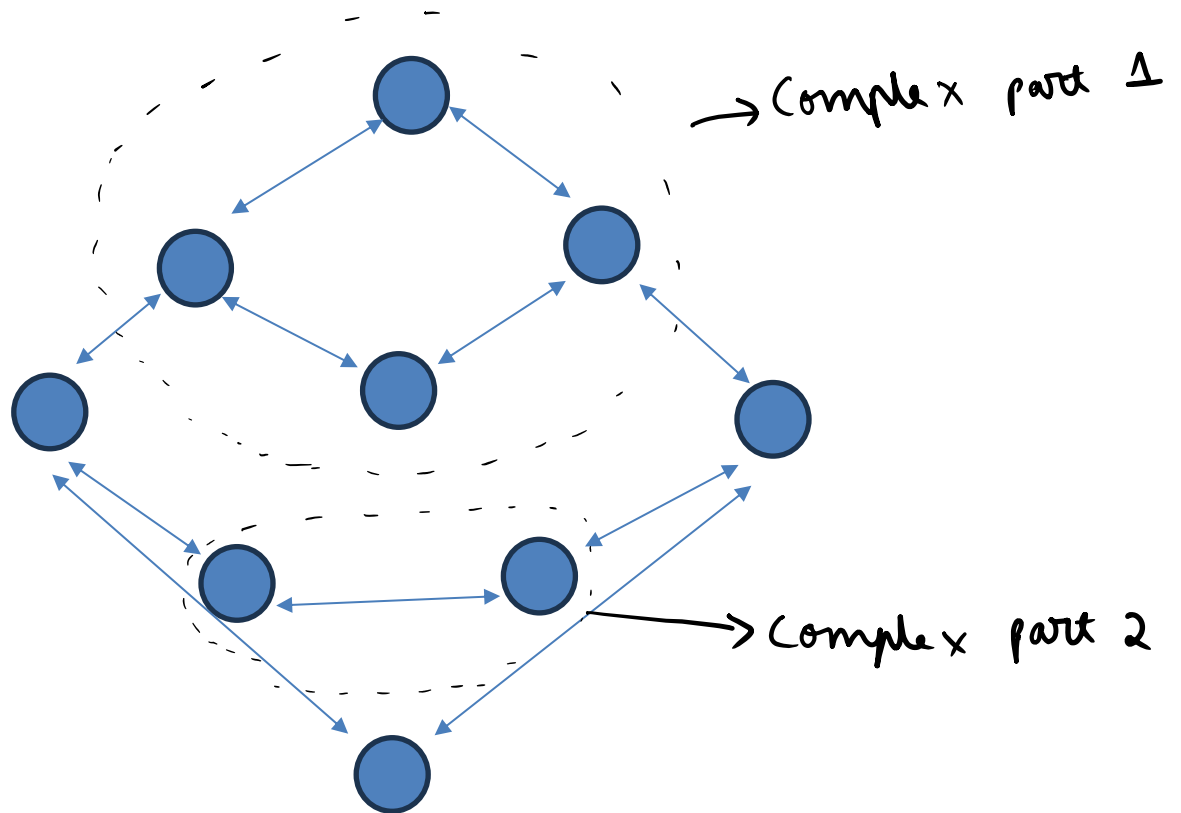
- Any combination of series-parallel circuits can be solved by using this algorithm.
 - Any circuit with a simple series parallel combination is easily solvable without marking any part as complex.
1. Structures having series-parallel configuration inside series parallel configuration must be marked as complex.

Eg. This is a simple series parallel circuit.

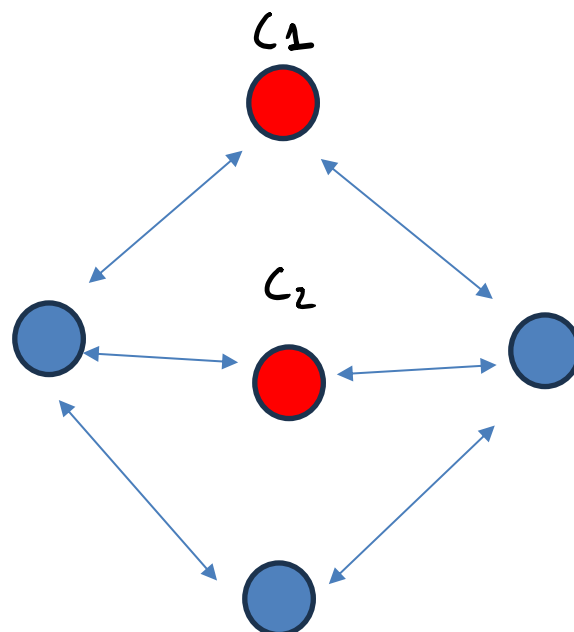


2. Marking systems as complex

Eg.



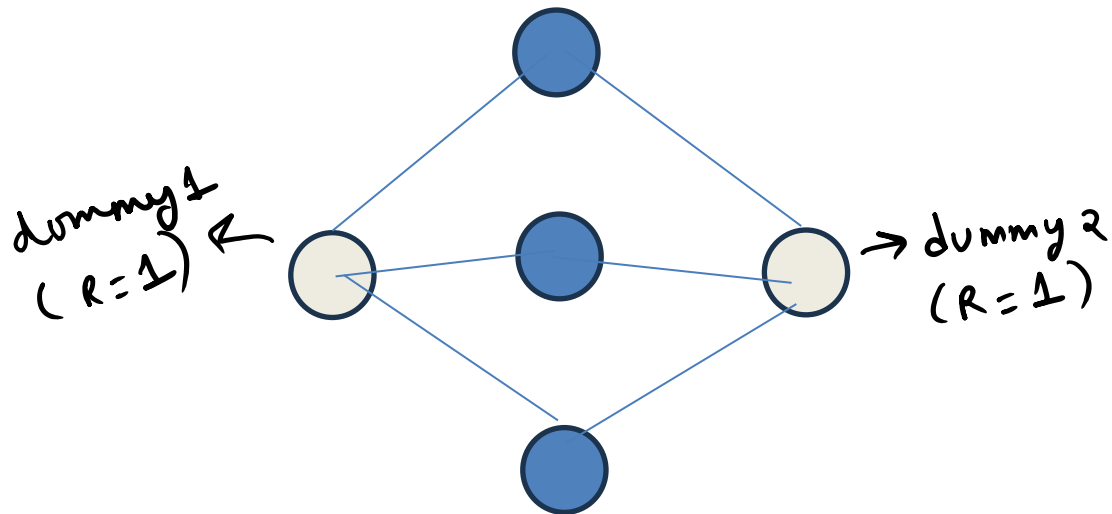
Reduced Diagram (while entering the nodes we must mark the node as complex parts for the algorithm to solve then the computer can infer it as a complex part and it reduces the diagram)



5.2.2 Limitations and challenges of the system:

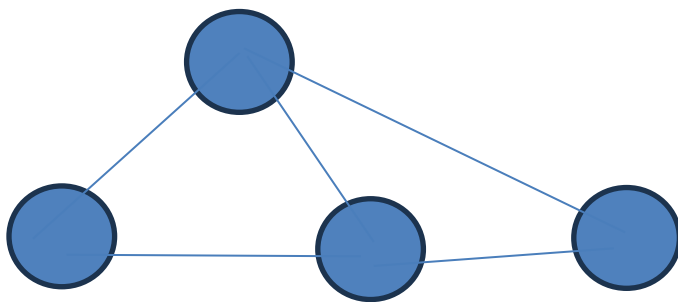
- User must mark complex regions manually and enter them as input.
- Circuits made by a combination of series parallel can only be solved.
- For circuit with has only parallel combinations dummy nodes must be inserted for system to recognize as parallel.

Eg: 3 nodes in parallel



- Configurations with stars cannot be solved

Eg:



Chapter 6 - Demonstration of web tool using sample data

Taking sample failure data for [513 ,649, 740, 814, 880, 944, 1004, 1078, 1161, 1282] and testing for the tool.

6.1 Life data features

6.1.1 Entering data

The user enters the data on the selected data type which is exact data.

The screenshot displays a web application interface for entering exact failure data. At the top, there is a navigation bar with links: Home, About, Build Your Simulation Model, and Contact. On the right side of the navigation bar, there is a user greeting 'Hello suraj!' and a 'Logout' button. The main content area is titled 'Exact Data' and contains a table with 14 rows. The table has three columns: 'SR NO', 'COMPONENT ID', and 'FAILURE TIME'. The first 10 rows are pre-filled with the failure times: 513, 649, 740, 814, 880, 944, 1004, 1078, 1161, and 1282. The remaining 4 rows (11, 12, 13, 14) are empty for user input.

SR NO	COMPONENT ID	FAILURE TIME
1		513
2		649
3		740
4		814
5		880
6		944
7		1004
8		1078
9		1161
10		1282
11		
12		
13		
14		

Figure 7 Exact data input frontend

6.1.2 Distribution selection and ranking

Select Required Distributions	Distribution Ranking																
<div>2P Weibull Distribution</div> <div>Exponential Distribution</div> <div>Gamma Distribution</div> <div>Normal Distribution</div> <div>Lognormal Distribution</div> <div>Loglogistic Distribution</div> <div>Gumbel Distribution</div> <div>3P Weibull Distribution</div> <div>Submit</div>	<table><thead><tr><th>Ranking</th><th>Distribution Name</th></tr></thead><tbody><tr><td>1</td><td>Normal Distribution</td></tr><tr><td>2</td><td>Lognormal Distribution</td></tr><tr><td>3</td><td>Gumbel Distribution</td></tr><tr><td>4</td><td>2P Weibull Distribution</td></tr><tr><td>5</td><td>Gamma Distribution</td></tr><tr><td>6</td><td>Loglogistic Distribution</td></tr><tr><td>7</td><td>Exponential Distribution</td></tr></tbody></table>	Ranking	Distribution Name	1	Normal Distribution	2	Lognormal Distribution	3	Gumbel Distribution	4	2P Weibull Distribution	5	Gamma Distribution	6	Loglogistic Distribution	7	Exponential Distribution
Ranking	Distribution Name																
1	Normal Distribution																
2	Lognormal Distribution																
3	Gumbel Distribution																
4	2P Weibull Distribution																
5	Gamma Distribution																
6	Loglogistic Distribution																
7	Exponential Distribution																

Figure 8- distribution ranking page

6.1.3 Parameters page

DISTRIBUTION NAME	P1 (ALPHA,MU,LAMBDA)	P2 (BETA,SIGMA,GAMMA)	P3 (GAMMA)
Normal Distribution	906.5	224.549	--
Lognormal Distribution	6.776	0.265	--
Gumbel Distribution	1017.734	205.246	--
2P Weibull Distribution	993.556	4.61	--
Gamma Distribution	59.844	15.148	--
Loglogistic Distribution	892.898	6.5	--
Exponential Distribution	0.003	513.0	--

Figure 9 Parameters page

6.1.4 Quick calculation pad

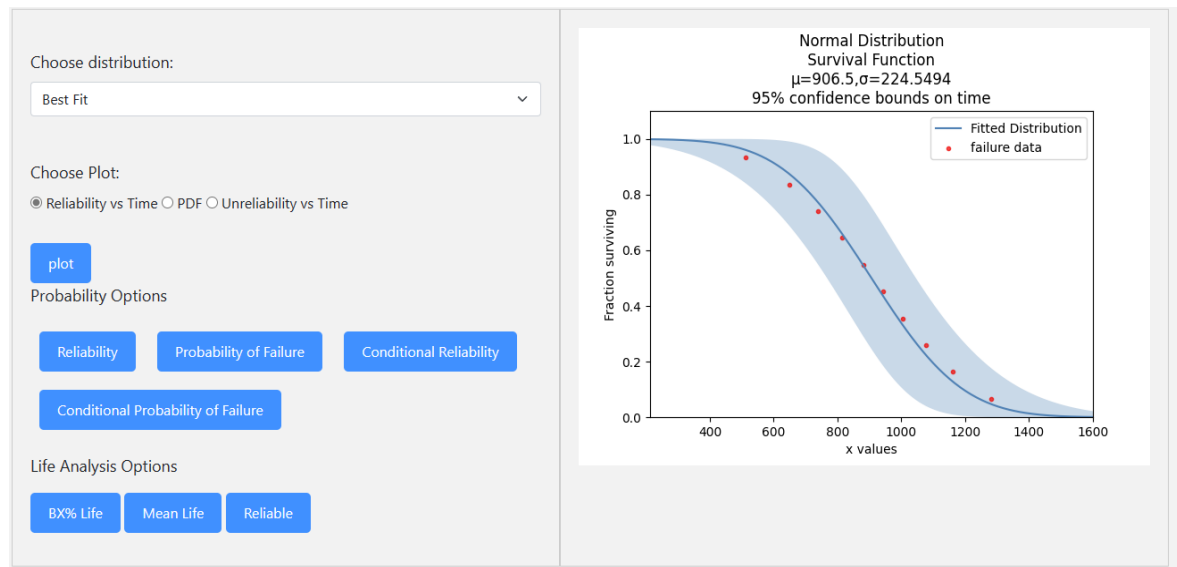


Figure 10 Qcp-plots

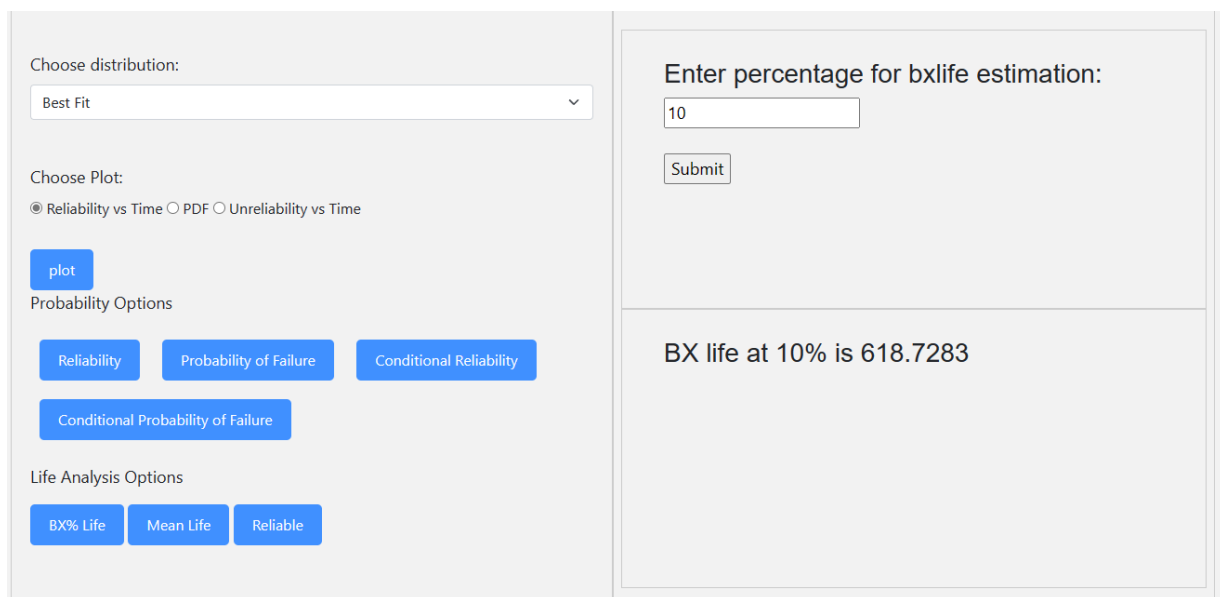


Figure 11 Qcp-Bx life

Choose distribution:

Best Fit

Choose Plot:

☒ Reliability vs Time
 ☐ PDF
 ☐ Unreliability vs Time

plot

Probability Options

Reliability

Probability of Failure

Conditional Reliability

Conditional Probability of Failure

Life Analysis Options

BX% Life

Mean Life

Reliable

Enter time for reliability estimation:

760

Submit

Reliability at time 760 is 74.2934 %

Figure 12 Qcp-reliability

Choose distribution:

Best Fit

Choose Plot:

☒ Reliability vs Time
 ☐ PDF
 ☐ Unreliability vs Time

plot

Probability Options

Reliability

Probability of Failure

Conditional Reliability

Conditional Probability of Failure

Life Analysis Options

BX% Life

Mean Life

Reliable

Enter present age of component for mean life to failure estimation:

700

Submit

Mean life to failure is 277.9788

Figure 13 Qcp mean-life

6.2 Block sim backend features

6.2.1 Entering node data

User enter the number of nodes and the distribution and the parameters of corresponding distribution.

```
Enter name of system:s1
Enter Time to calculate reliability50
Enter details of system s1
Enter the number of nodes20
Enter node details of node 1
Enter name:b1
Is this block a node or complex_part or subsystem( 0 or 1 or 2?)0
Enter distribution name:a
Enter Eta value:100
Enter Beta value:2.5
Enter node details of node 2
Enter name:b2
Is this block a node or complex_part or subsystem( 0 or 1 or 2?)0
Enter distribution name:a
Enter Eta value:100
Enter Beta value:2.5
```

Figure 14 Block sim Terminal input

6.2.2 Define connections number and pair wise names of blocks

```
Enter number of connections20
Type connections pair wise:
b1 c1
pair0 b1
pair1 c1
c1 b7
pair0 c1
pair1 b7
b1 c2
```

Figure 15 Block sim connections

6.2.3 Output system at particular time.

```
The system reliability at time t = 50 0.6648  
0.6648
```

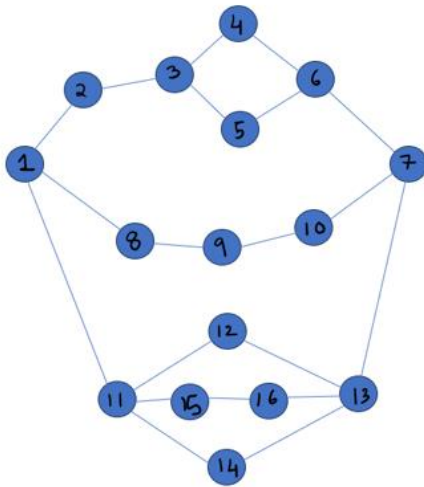
Chapter 7 - Conclusion and future scope

7.1 Results

Comparing the Results of the life data features with the actual software are fairly accurate error of about 6-7% hence the tool can be used to estimate reliability in industries by a considering a small offset for the error. The compared result can be seen below tested for the sample data used in the webtool demonstration.

Parameter	Weibull++	Webtool
Reliability@ 200hrs	99.84%	99.93%
B10 life	570 hrs	610 hrs
MTTF	906.34 hrs	908.51 hrs
Reliable life at 50%	900hrs	918.18hrs
Parameters	Beta=4.002, Eta=999.91	Beta=4.61, Eta=994.122

In case of RBDs the results were tested for some of the configurations



Total Nodes:
16(simple)+4(complex)

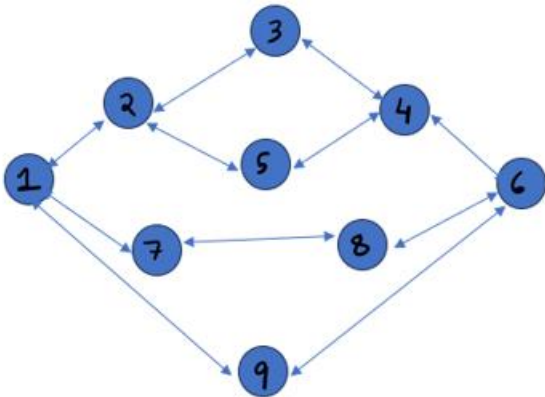
Complex Nodes: 4

No of connections: 20

Calculated system reliability: 0.6648

Actual Reliability: 0.6648

Figure 16 RBD config-1



Total Nodes: 9(simple)+2(complex)
Complex Nodes: 2
No of connections: 11

Calculated system reliability: 0.6915

Actual Reliability: 0.691532

Figure 17RBD config -2

7.2 Future Scope

- To develop a designated frontend for the multicomponent systems features where users can freely drag and drop blocks to create the RBD and display the results.
- Eliminating the need to mark components as complex which would eliminate the human intervention for solving the circuit.
- Integration of the block sim features to the simuli-ware software and deployment of the code on cloud platforms such as AWS, Azure etc.

References

- Cousineau, D., 2009. Fitting the three-parameter Weibull distribution: Review and evaluation of existing and new methods. *IEEE Transactions on Dielectrics and Electrical Insulation*, 16(1), pp.281-288.
- Bhattacharya, P. and Bhattacharjee, R., 2010. A study on Weibull distribution for estimating the parameters. *Journal of Applied Quantitative Methods*, 5(2), pp.234-241.
- Lyu, M.R., 2007, May. Software reliability engineering: A roadmap. In *Future of Software Engineering (FOSE'07)* (pp. 153-170). IEEE.